**Physical Design**

A. Table Stats for Each Table

**review\_table :**

Total = 249 Bytes

*Computation of the Blocking Factor for the table using the standard block size of 512 bytes.*

Blocking factor = 512/249 = 2 Records per block.

Number of file blocks = 3,00,000/2 = 150000 blocks

**Storage Requirements:**

Number of bytes waster per block= 249 x 2=498

Percentage wastage = (512-498/512) x 100 = (14/512) x 100 = 2.734%

Hence **unspanned** file type is used

Each record effectively occupies (512/2) = 256 bytes

* Overhead of 7 bytes per record.

**watch\_list :**

Total = 29 Bytes

*Computation of the Blocking Factor for the table using the standard block size of 512 bytes.*

Blocking factor = 512/29 = 17 Records per block.

Number of file blocks = 3,00,000/17 = 17648 blocks

**Storage Requirements:**

Number of bytes waster per block= 29 x 17=493

Percentage wastage = (512-493/512) x 100 = (19/512) x 100 = 3.71%

Hence **unspanned** file type is used

Each record effectively occupies (512/17) =30.118 bytes

* Overhead of 1.118 bytes per record.

**user\_table :**

Total = 46 Bytes

*Computation of the Blocking Factor for the table using the standard block size of 512 bytes.*

Blocking factor = 512/46 = 11 Records per block.

Number of file blocks = 3,00,000/11 = 27272 blocks

**Storage Requirements:**

Number of bytes waster per block= 46 x 11=506

Percentage wastage = (512-506/512) x 100 = (6 /512) x 100 = 1.17%

Hence **unspanned** file type is used

Each record effectively occupies (512/11) = 46.54 bytes

* Overhead of 0.54 bytes per record.

**rating\_table :**

Total = 33 Bytes

*Computation of the Blocking Factor for the table using the standard block size of 512 bytes.*

Blocking factor = 512/33 = 15 Records per block.

Number of file blocks = 3,00,000/15 = 20000 blocks

**Storage Requirements:**

Number of bytes waster per block= 33 x 15=495

Percentage wastage = (512-495/512) x 100 = (17/512) x 100 = 3.32%

Hence **unspanned** file type is used

Each record effectively occupies (512/15) = 34.133 bytes

* Overhead of 1.133 bytes per record.

**movie\_artist\_award\_table :**

Total = 41 Bytes

*Computation of the Blocking Factor for the table using the standard block size of 512 bytes.*

Blocking factor = 512/41 = 12 Records per block.

Number of file blocks = 3,00,000/12 = 25000 blocks

**Storage Requirements:**

Number of bytes waster per block= 41 x 12=492

Percentage wastage = (512-492/512) x 100 = (10/512) x 100 = 1.9531%

Hence **unspanned** file type is used

Each record effectively occupies (512/12) = 42.66 bytes

Overhead of 1.666 bytes per record.

**genre\_table :**

Total = 150 Bytes

*Computation of the Blocking Factor for the table using the standard block size of 512 bytes.*

Blocking factor = 512/150 = 3 Records per block.

Number of file blocks = 3,00,000/3 = 100000 blocks

**Storage Requirements:**

Number of bytes waster per block= 150 x 3=450

Percentage wastage = (512-450/512) x 100 = (62/512) x 100 = 12.109%

Hence **spanned** file type is used

Each record effectively occupies (512/3) = 170.666 bytes

* Overhead of 20.666 bytes per record.

**sequence\_table :**

Total = 8 Bytes

*Computation of the Blocking Factor for the table using the standard block size of 512 bytes.*

Blocking factor = 512/8 = 64 Records per block.

Number of file blocks = 3,00,000/64 = 4688 blocks

**Storage Requirements:**

Number of bytes waster per block= 8 x 64=512

Percentage wastage = (512-512/512) x 100 = (0/512) x 100 = 0.0%

Hence **unspanned** file type is used

Each record effectively occupies (512/64) = 8 bytes

* Overhead of 0 bytes per record.

**movie\_artist\_table :**

Total = 8 Bytes

*Computation of the Blocking Factor for the table using the standard block size of 512 bytes.*

Blocking factor = 512/8 = 64 Records per block.

Number of file blocks = 3,00,000/64 = 4688 blocks

**Storage Requirements:**

Number of bytes waster per block= 64 x 8=512

Percentage wastage = (512-512/512) x 100 = (0) x 100 = 0%

Hence **unspanned** file type is used

Each record effectively occupies (512/64) = 8 bytes

* Overhead of 0 bytes per record.

**artist \_table :**

Total = 169 Bytes

*Computation of the Blocking Factor for the table using the standard block size of 512 bytes.*

Blocking factor = 512/169 = 3 Records per block.

Number of file blocks = 3,00,000/3 = 100000 blocks

**Storage Requirements:**

Number of bytes waster per block= 169 x 3=507

Percentage wastage = (512-507/512) x 100 = (5/512) x 100 = 0.9765%

Hence **unspanned** file type is used

Each record effectively occupies (512/3) = 170.66 bytes

* Overhead of 1.66 bytes per record.

**Access Method:**

1. Binary search on this data file = log2(100000) = 17
2. Primary Indexing

Length of each index record = 4 bytes (Primary Key) + 5 bytes (Block Pointer) = 9 bytes

Index blocking factor = 512/9 = 57

Total number of entries in index file = Number of blocks = 100000

Therefore, number of blocks required 100000/57 = 1755 blocks

To perform binary search on index file log2(1755) = 11 block accesses

To search for a record we need 11+1 = 12 block accesses ( to access the data file block containing record).

**movie\_table :**

Total = 189 Bytes

*Computation of the Blocking Factor for the table using the standard block size of 512 bytes.*

Blocking factor = 512/189 = 2 Records per block.

Number of file blocks = 3,00,000/2 = 150000 blocks

**Storage Requirements:**

Number of bytes waster per block= 189 x 2=378

Percentage wastage = (512-378/512) x 100 = (134/512) x 100 = 26.17%

Hence **spanned** file type is used

Each record effectively occupies (512/2) = 256 bytes

* Overhead of 67 bytes per record.

**Access Method:**

1. Binary search on this data file = log2(150000) = 18
2. Primary Indexing

Length of each index record = 4 bytes (Primary Key) + 5 bytes (Block Pointer) = 9 bytes

Index blocking factor = 512/9 = 57

Total number of entries in index file = Number of blocks = 150000

Therefore, number of blocks required 150000 /57 = 2632 blocks

To perform binary search on index file log2(2632) = 12 block accesses

To search for a record we need 12+1 = 13 block accesses ( to access the data file block containing record).

**award\_table :**

Total = 169 Bytes

*Computation of the Blocking Factor for the table using the standard block size of 512 bytes.*

Blocking factor = 512/169 = 3 Records per block.

Number of file blocks = 3,00,000/3 = 100000 blocks

**Storage Requirements:**

Number of bytes waster per block= 169 x 3=507

Percentage wastage = (512-507/512) x 100 = (5/512) x 100 = 0.09765%

Hence **unspanned** file type is used

Each record effectively occupies (512/3) = 170.666 bytes

* Overhead of 1.666 bytes per record.

**Access Method:**

1. Binary search on this data file = log2(100000) = 17
2. Primary Indexing

Length of each index record = 4 bytes (Primary Key) + 5 bytes (Block Pointer) = 9 bytes

Index blocking factor = 512/9 = 57

Total number of entries in index file = Number of blocks = 100000

Therefore, number of blocks required 100000/57 = 1755 blocks

To perform binary search on index file log2(1755) = 11 block accesses

To search for a record we need 11+1 = 12 block accesses ( to access the data file block containing record).

B. Execution Diagram and Query Stats for Each Query

1. **Get movies**

1.1 film artist input

select jj.movie\_name, jj.genre, gg.suggested\_audience, jj.release\_date, jj.duration, jj.language, jj.rating, jj.num\_of\_ratings, jj.num\_of\_reviews

from

(select joined.movie\_name movie\_name, joined.genre genre, joined.release\_date release\_date, joined.duration duration, joined.language language, joined.rating rating, joined.num\_of\_ratings num\_of\_ratings, joined.num\_of\_reviews num\_of\_reviews from

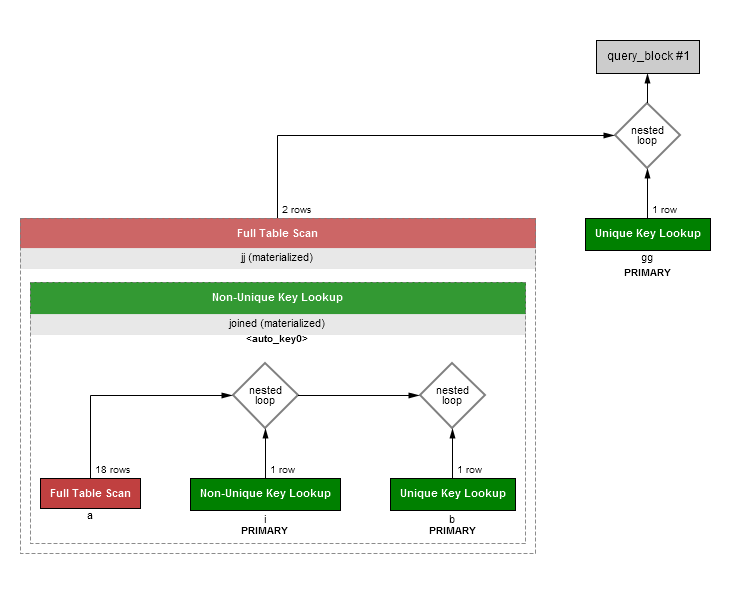
(select a.mname movie\_name, a.duration duration, a.language language, a.genre genre, a.release\_date release\_date, a.rating rating, a.num\_of\_ratings num\_of\_ratings, a.num\_of\_reviews num\_of\_reviews, b.art\_fname fname, b.art\_lname lname

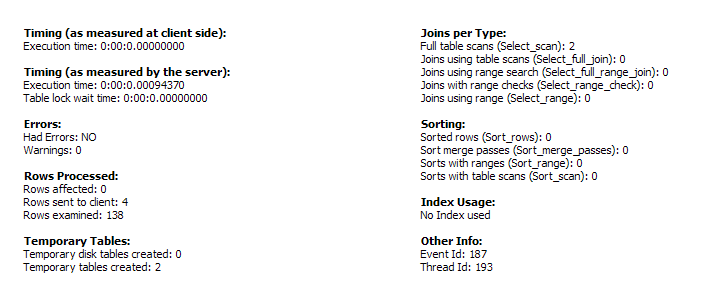
from movie\_artist\_table i, movie\_table a, artist\_table b

where i.movie\_id = a.movie\_id and b.artist\_id = i.artist\_id) joined

where joined.fname = 'INPUT.fname' and joined.lname = 'INPUT.lname') jj, genre\_table gg

where jj.genre = gg.genre\_name;



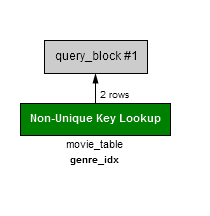


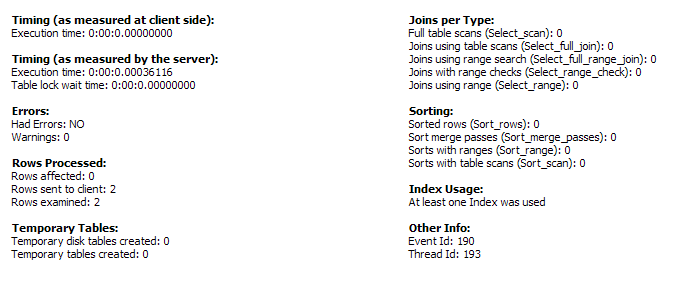
1.2 INPUT genre

select mname, release\_date, duration, language

from movie\_table

where genre = 'INPUT.genre';





1.3 INPUT award

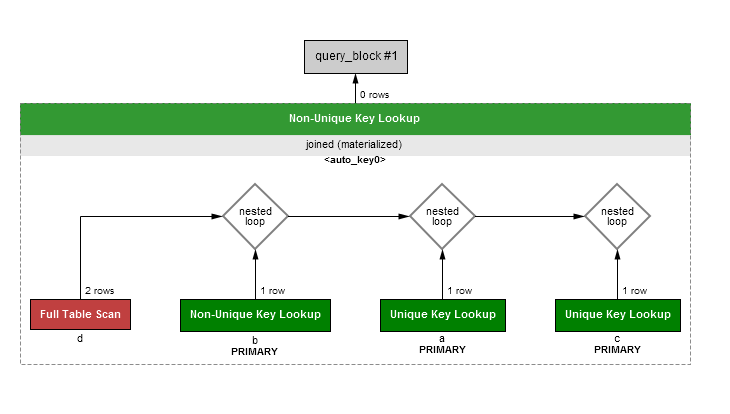
select joined.movie\_name, joined.duration, joined.language, joined.genre, joined.release\_date, joined.category, joined.year, joined.fname, joined.lname, joined.rating, joined.num\_of\_ratings, joined.num\_of\_reviews

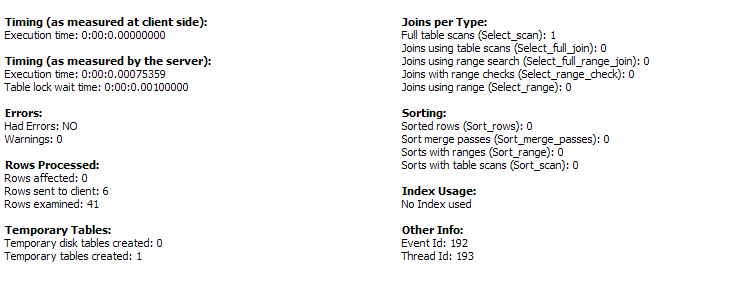
from (select a.mname movie\_name, a.duration duration, a.language language, a.genre genre, a.release\_date release\_date , b.category category, b.year year, c.art\_fname fname, c.art\_lname lname, d.award\_name award, a.rating rating, a.num\_of\_ratings num\_of\_ratings, a.num\_of\_reviews num\_of\_reviews

from movie\_table a, movie\_artist\_award\_table b, artist\_table c, award\_table d

where a.movie\_id=b.movie\_id and b.artist\_id=c.artist\_id and b.award\_id=d.award\_id) joined

where joined.award='INPUT.award\_name';



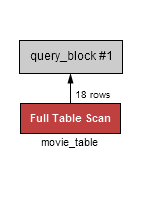


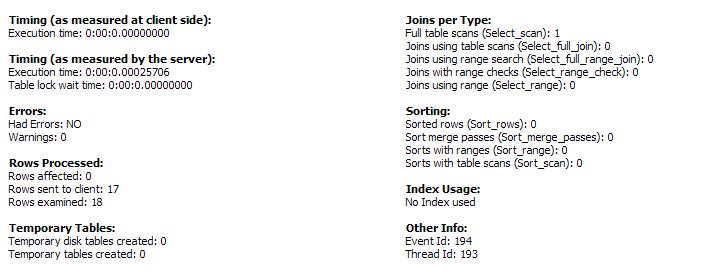
1.4INPUT language

select mname, release\_date, duration, genre, rating, num\_of\_ratings, num\_of\_reviews

from movie\_table

where language = 'INPUT.language';



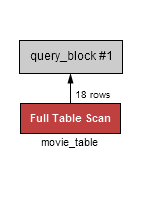


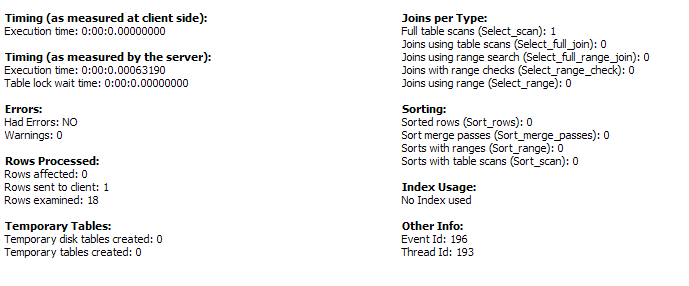
1.5INPUT year

select mname, duration, genre, language, rating, num\_of\_ratings, num\_of\_reviews

from movie\_table

where release\_date = 'INPUT.year';





2. **GET ARTIST**

2.1INPUT movie name

select artist\_table.art\_fname, artist\_table.art\_lname, artist\_table.job\_title, artist\_table.age, artist\_table.origin

from movie\_artist\_table

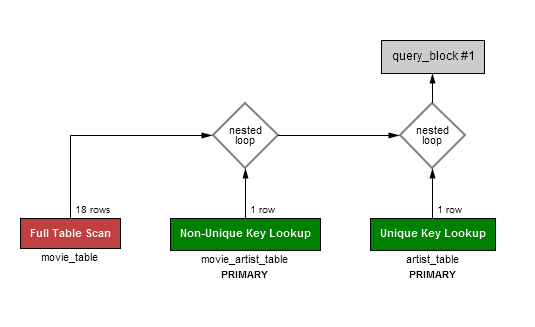
inner join artist\_table

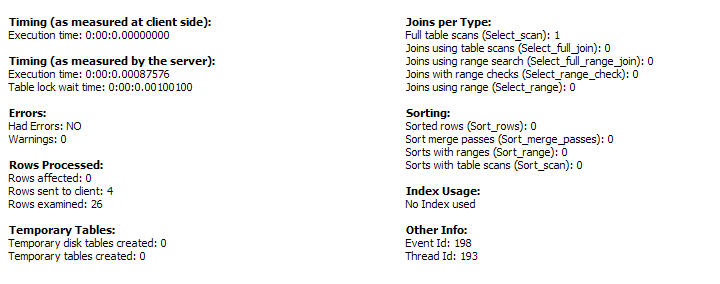
on artist\_table.artist\_id=movie\_artist\_table.artist\_id

inner join movie\_table

on movie\_table.movie\_id=movie\_artist\_table.movie\_id

where movie\_table.mname='INPUT.mname';





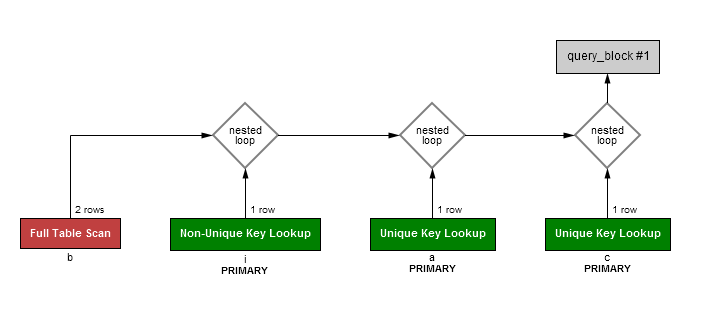
2.2 INPUT award name

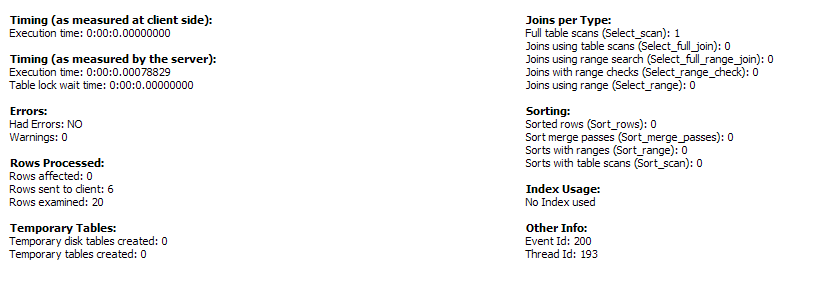
select c.art\_fname, c.art\_lname, i.category, i.year, c.job\_title, c.age, c.origin, a.mname, a.genre, a.language, a.release\_date, a.duration, a.rating, a.num\_of\_ratings, a.num\_of\_reviews

from movie\_artist\_award\_table i, movie\_table a, award\_table b, artist\_table c

where i.movie\_id=a.movie\_id and i.award\_id=b.award\_id and i.artist\_id=c.artist\_id

and b.award\_name='INPUT.award\_name';





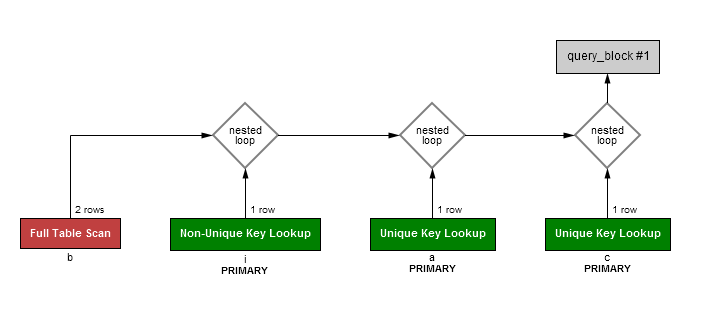
2.3.special: award and genre

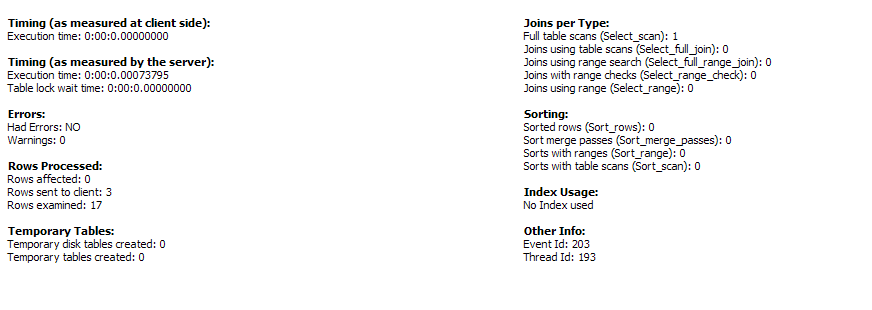
select c.art\_fname, c.art\_lname, i.category, i.year, c.job\_title, c.age, c.origin, a.mname, a.genre, a.language, a.release\_date, a.duration, a.rating, a.num\_of\_ratings, a.num\_of\_reviews

from movie\_artist\_award\_table i, movie\_table a, award\_table b, artist\_table c

where i.movie\_id=a.movie\_id and i.award\_id=b.award\_id and i.artist\_id=c.artist\_id

and b.award\_name='INPUT.award\_name' and a.genre='INPUT.genre';

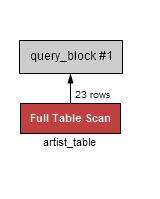




2.4 INPUT job title

select art\_fname, art\_lname, age, origin from artist\_table

where job\_title=INPUT;





3. **GET AWARD**

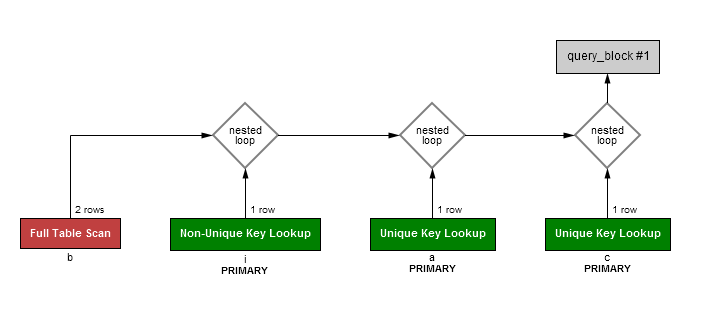
3.1 input artist fname and lname

select b.award\_name, i.category, i.year, b.organisation, b.description, a.mname, a.genre, a.rating

from movie\_artist\_award\_table i, movie\_table a, award\_table b, artist\_table c

where i.movie\_id=a.movie\_id and i.award\_id=b.award\_id and i.artist\_id=c.artist\_id

and c.art\_fname = 'INPUT.fname' and c.art\_lname = 'INPUT.lname';





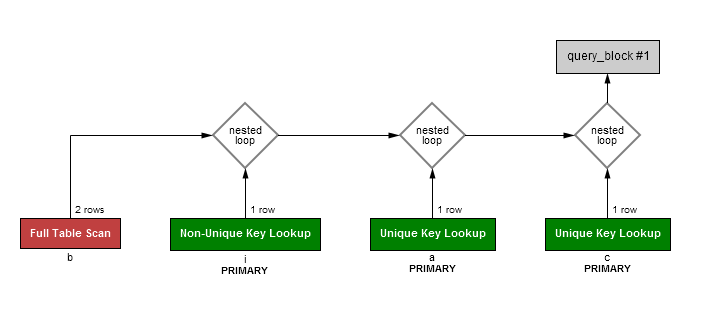
3.2 award by movie

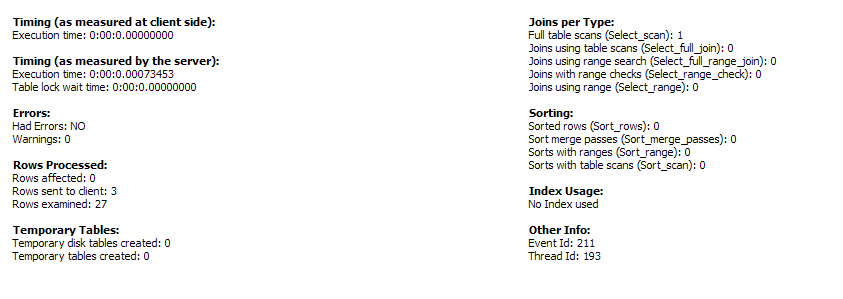
select b.award\_name,i.category, i.year, b.organisation, b.description, c.art\_fname, c.art\_lname, c.job\_title, c.age, c.origin

from movie\_artist\_award\_table i, movie\_table a, award\_table b, artist\_table c

where i.movie\_id=a.movie\_id and i.award\_id=b.award\_id and i.artist\_id=c.artist\_id

and a.mname = 'INPUT.movie\_name';





4. **Get TOP MOVIES**

4.1 top 3 by genre

select \* from

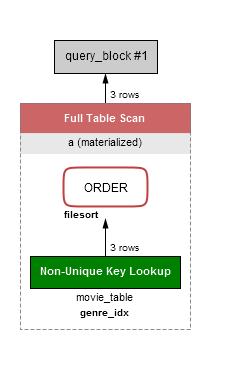
(select mname, duration, language, release\_date, rating, num\_of\_ratings, num\_of\_reviews

from movie\_table

where genre='INPUT.genre'

order by rating desc) a

limit 3;





4.2top 3 by film actor

select \* from

(select \* from

(select movie\_table.mname, movie\_table.duration, movie\_table.language, movie\_table.release\_date, movie\_table.genre, movie\_table.rating rating

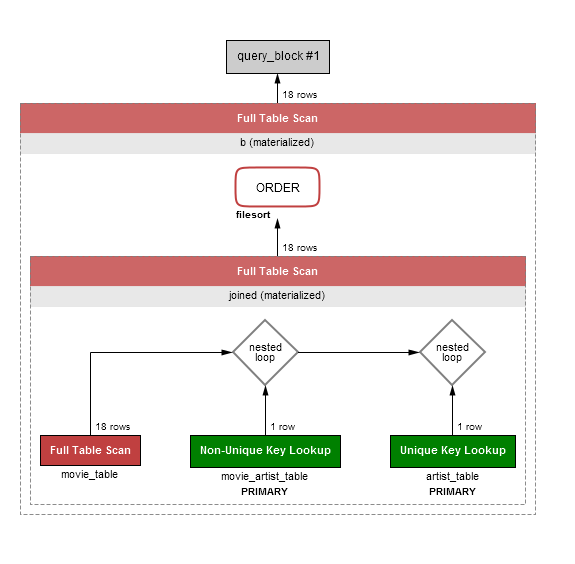
from movie\_artist\_table, artist\_table, movie\_table

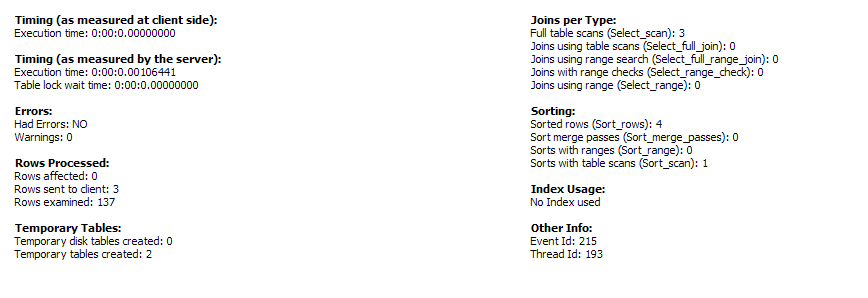
where artist\_table.artist\_id=movie\_artist\_table.artist\_id and movie\_table.movie\_id=movie\_artist\_table.movie\_id

and artist\_table.art\_fname='INPUT.fname' and artist\_table.art\_lname='INPUT.lname') joined

order by joined.rating desc) b

limit 3;





4.3 top 3

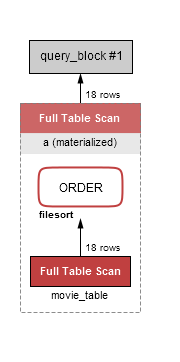
select \* from

(select mname, duration, genre, language, release\_date, rating

from movie\_table

order by rating desc) a

limit 3;





6.number of movies contributed each a film artist

select aa.art\_fname, aa.art\_lname, jj.num\_of\_movies, aa.job\_title, aa.origin, aa.age

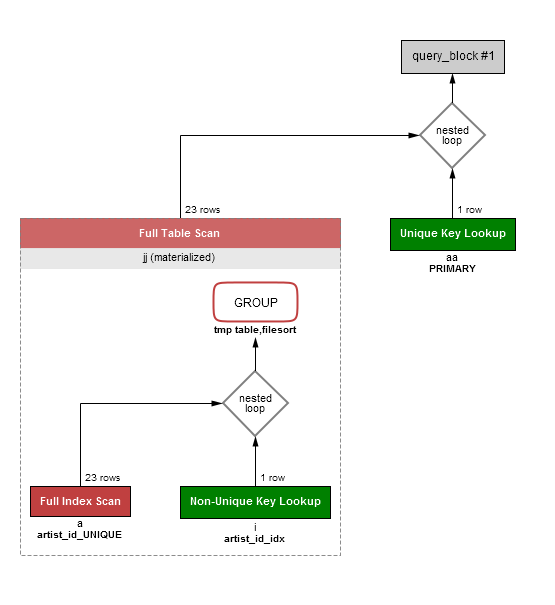
from (select i.artist\_id, count(\*) num\_of\_movies

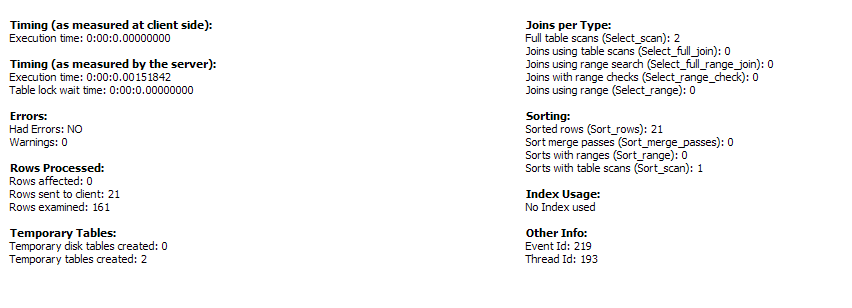
from movie\_artist\_table i, artist\_table a

where i.artist\_id = a.artist\_id

group by artist\_id) jj, artist\_table aa

where jj.artist\_id=aa.artist\_id;





C. Query Block Access Time Calculation

1. Calculating disk acces time for 1 block:

Size of a block: 512 bytes/sector

Seek time: 12 ms

Disk rotation speed: 5400 rpm

Data Transfer speed: 4 mbps

Overhead = 1ms

Queing delay: 0 s

* 12 + 1/5400 (1/rpm) + 512/4000000 (1/b/s) +1 (ms)
* 12 + ( 1/90(1/rps) \* 1000) + 512/4000 (1/b/ms) +1 (ms)
* 12 + 11 + 0.1 + 1
* 24.1 ms

Considering average rotational delay, we get

* 12 + ( 1/(90 \* 2)(1/rps) \* 1000) + 512/4000 (1/b/ms) +1 (ms)
* 12 + 5.5 + 0.1 + 1
* 18.6 ms

1. Consider the following Search query:

We have movies and ratings for each movies:

*“select \* from movie\_table where movie\_id=17;”*

Considering 300000 entries.

size of 1 movie entry=189 bytes.

Blocking factor=floor(512/189)=2 => 2 records per block.

(Assuming others as wastage since we consider unspanned file storage type.)

So no. of blocks required=ceiling(300000/2)=150000 blocks.

We consider time calculation for following 4 cases:

Case 1:

No ordering based on any field.

Linear search will take O(n/2) block accesses,

so approximately 150000/2 =75000 block accesses.

* So total time taken for query = 75000 \* 18.6 = 1395000ms

Case 2:

Ordering based on rating,

so binary search would require ceiling(log(150000) to the base 2)+1=18+1=19 block accesses.

* So total time taken for query = 19 \* 18.6 = 353.4 ms

Case 3:

Index based on **movie\_id**,

Size of movie\_id=4 bytes, block ptr size = 6 bytes.

So size of index entry=4+6=10 bytes.

So no. Of index entries per block=floor(512/10)=51 entries.

Total no. Of index entries = total no. Of blocks in data file = 150000

so index blocking factor(no. Of blocks required to store index entries, b\_fact)=

ceiling( 150000/51)=2941 entries

so now search will take ceiling(log(2941) to base 2)+1=11+1=12 block accesses

* So total time taken for query = 12 \* 18.6 = 223.2 ms

Case 4:

*Multilevel* index on **movie\_id**,

1st level index(b\_fact) = 2941 entries

2nd level index = ceiling(2941 /51) = 58 entries

3rd level index = ceiling(58/51)=1 entries

So no. Of block accesses = 3(level of indexing) + 1=4 block accesses.

* So total time taken for query = 4 \* 18.6 = 74.4 ms